implementation for which a package has been modeled. Programs can be dynamically constructed at runtime by accessing the various packages and models and performing operations. Since there exists a single set of interfaces defining the API to call any method or construct any component, it becomes relatively easy to build applications that use a CIE to construct dynamic programs at runtime.

[0413] A persistence engine is necessary for a component integration engine that wishes to store data related to instances created or dynamic programs to be stored for future use. The persistence engine can be as simple as saving the assemblies to files as XML or it may be as complex as a full Object-Relational Mapping Engine (see details in section below). The persistence engine uses metamaps (a mapping of one meta-implementation to another meta-implementation) to determine how to persist the given instance. A broker that performs the actual work of storing or retrieving the object interprets these maps. Like all other parts of a CIE, the persistence engine enforces access and security constraints based on authorizations related to the authentication of the user.

[0414] Providing commonly used services as components in the component integration engine increases the reuse of that code and reduces the amount of code necessary to create new applications. A common pattern for providing these services is to create command architectures. Each command takes a set of input and produces output. With a component integration engine, commands become much more powerful and flexible. Since each component has the metamodel layer to describe the structure of its input and output, commands can be created with much more complex input and output models while still being understandable to the user. Commands are also more configurable through the use of the meta-implementation layer, allowing generalized commands to be configured to perform specific tasks.

[0415] The services described below are examples of the types of services a component integration engine may provide. These do not represent all possible services. Like all other parts of a CIE, the common services enforce access and security constraints based on authorizations related to the authentication of the user.

[0416] This service provides access to the objects in the environment in which the component integration engine is running. It provides the ability to read the configuration of any shared object or service in the environment, to add or remove these objects or services, or reconfigure these objects and services.

[0417] A database service provides access to databases and other database-like data stores like flat-files, fixed-length files, and spreadsheets. Commands may exist to execute SQL, store and retrieve data, and alter the relational structures in the database. Database specific commands may allow configuration of the database environment.

[0418] A domain service provides access to the domains that exist in the component integration engine. Commands may exist to start, suspend, resume, stop, or dispose of services and shared objects. Commands may exist to create new virtual hosts and domains. Used together with the configuration service, the domain service provides complete control over the environment in which the CIE resides.

[0419] An email service provides access to email servers. Commands may exist to send and receive email messages, manage mail stored on an email server, add or remove users from message lists, and add or remove users from the email server.

[0420] A formatting service provides formatting of textual information. Commands may exist to process mail-merge templates, XSLT transformations, or tag libraries.

[0421] An image service provides access to image manipulation and creation activities. Commands may exist to create images from GUI components, create images based on XML descriptors, or draw text as images.

[0422] A messaging service provides access to messaging services. Messaging services guarantee delivery of a message to those parties that have registered interest in receiving messages or to those parties to whom the message is addressed.

[0423] A naming service provides access to directory and naming context servers. Commands may exist for user lookup, user password verification, computer registry access and modification, and phone book lookups.

[0424] An object store service converts an object structure to a state where it will exist even after the object has been destroyed. Commands may provide object-relational mapping, Enterprise Java Beans, or XML file creation.

[0425] A processing service allows the creation of new commands or services through the creation of a process control structure. This control structure may contain loops and branches and other operations available as part of the CIE. When a user requests the execution of a specific process, it is executed like a virtual program. Security restrictions determine the ability of users or programs to create new processes.

[0426] A resource service provides access to files, URLs, database LOBS, and other sources of binary and textual data. Resources serve as a persistence mechanism for binary and textual data. Commands may exist to create new resources, delete existing resources, or read the content of a resource.

[0427] A roadmap service allows the creation of new commands or services through the straightforward combination of other existing commands. A roadmap specifies each step in a process in the order in which it should occur. The commands in this service execute the roadmap.

[0428] A schedule service provides for the scheduling of activities on a one-time or reoccurring basis. Commands exist to schedule activities and reminders for these activities for any desirable interval and reoccurrence. Commands also exist to view existing activities and remove activities.

[0429] A security service provides access to the system security settings. Normally, security is read-only and restricted to those users with appropriate access. Sometimes it is useful to have access to commands that retrieve the current user's identity, so it can be passed as parameters to other commands (like logging).

[0430] Several very difficult to implementation concepts become very simple when a component integration engine is applied to them. Described below are some of these implementations and the manner in which a component integration engine simplifies them.